Object Dynamics Modeling with Hierarchical Point Cloud-based Representations

Supplementary Material

In this document, we provide more details about our method, which is presented in the main paper, as well as additional experimental results.

1. Finding Interaction Points

Here, we describe an approach used for identifying interaction points between mesh faces of different objects. Initially, we sample points from mesh faces uniformly and calculate the distances between these sampled points across the mesh faces of different objects. Note that a pair of mesh faces can have multiple pairs of sampled points, given that each mesh face can keep multiple sampled points. Interaction point pairs are identified when the distance between sampled points on different faces falls below a specified threshold. Subsequently, any pair of mesh faces containing at least one interaction point pair is considered to have a distance approximately within the threshold. Note that our proposed Relational PointConv, designed for processing mesh data directly, is also compatible with other approaches for identifying closely positioned mesh faces, such as the Bounding Volume Hierarchy algorithm used in [1].

2. Speed Comparison

We ran SGNN's [4] and our inference code with the same point cloud input on the same machine with RTX 3090. Averaging across the Physion scene types, our inference code ran 17.10 fps while SGNN's code ran 38.9 fps. The SGNN code is built based on the DPI code, so we expect DPI to run at a similar speed. Our implementation is still fast, given DPI and SGNN are shallow networks with only two layers of hierarchy.

3. Point Cloud Generation

For both datasets, ground truth meshes are available. Hence, we uniformly sample points from the mesh faces to obtain a point cloud dense on object surfaces. Then, we employ an initial grid sampling step with a voxel size of 0.05 for Physion and 0.2 for Kubric. This process subsamples the point cloud with one point per voxel. Afterward, within the U-Net architecture, we utilize different voxel sizes for downsampling at different levels. For Physion, we employ voxel sizes of 0.075, 0.1125, and 0.16875 at different downsampling levels. For Kubric, we use voxel sizes of 0.3, 0.45, and 0.675 at different downsampling levels.

For the distance threshold r used in KNN for relational PointConv, we select 0.1 as the threshold at the highest resolution in U-Net for Physion. Subsequently, for the downsampling levels, we employ thresholds of 0.15, 0.225, and 0.3375. Regarding Kubric, we utilize a threshold of 0.4 at the highest resolution, with thresholds of 0.6, 0.9, and 1.35 for the downsampling levels.

4. Mesh Preprocessing

We use the Kubric dataset for our mesh experiments. Since the original meshes in the Kubric dataset come with too many vertices, we first perform mesh simplification via vertex clustering with a voxel size of 0.2 and 0.4 for Movi-A and Movi-C, respectively. For the downsampling layers in the U-Net architecture, we adopt the mesh simplification method proposed in [3] instead of the grid downsampling used for point clouds.

5. Interaction PointConv U-Net for Mesh

We used a variant of the PointConv U-Net architecture shown in Fig. 1 for our mesh experiments. This choice was made because applying relational PointConv to downsampled meshes may result in inaccurate face-to-face collision modeling unless the downsampling process accurately preserves the shapes of object surfaces.

6. Training details

We utilized the Adam optimizer with an initial learning rate of 0.001 and trained the model for 30 epochs for the Physion dataset. For the Kubric dataset, we employed an initial learning rate of 0.005 and trained the model for 15 epochs. During each training iteration, a single input frame was randomly selected from each training video. We defined one epoch as the point when every video in the training dataset is sampled N times for input frames. In the case of Physion, we set N to 8, consistent with previous studies [2, 4], while for Kubric, we set N to 2.

7. Additional Ablation Results

In this section, we present results from additional ablation experiments that we conduct with acceleration prediction models. In Table 1, we compare the performance using different numbers of interaction blocks used in the U-Net. For this experiment, we use the same number of interaction blocks in the encoder, bottleneck, and decoder of U-Net for simplicity, but note that a different number of blocks can be used for the bottleneck in practice depending on the needs

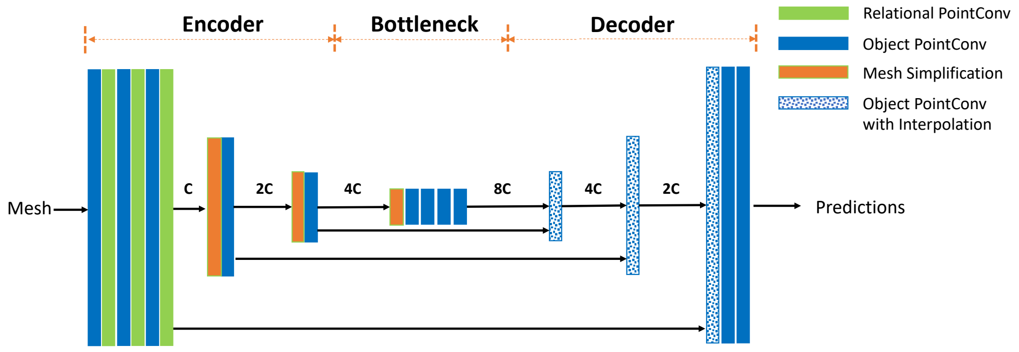

Figure 1. The proposed U-Net architecture for mesh inputs. Unlike its counterpart designed for dense point cloud inputs, we only utilize relational PointConv at the highest resolutions, as some object shapes might not be accurately represented with fewer mesh vertices. In this U-net architecture, the downsampling layers only have object PointConv that propagates collision effects over the mesh vertices of each object.

	Dominoes	Contain	Link	Support	Drop	Collide	Roll	Average
3 blocks (1 encoding, 1 bottleneck, 1 decoding)	90.2 ± 1.1	72.2 ± 3.6	73.1 ± 4.1	83.3 ± 0.9	85.8 ± 2.2	91.1 ± 1.4	87.5 ± 0.3	83.3
6 blocks (2 encoding, 2 bottleneck, 2 decoding)	88.7 ± 1.6	75.8 ± 1.7	74.2 ± 0.8	83.8 ± 1.7	87.4 ± 0.9	91.6 ± 0.6	87.3 ± 0.0	84.1
9 blocks (3 encoding, 3 bottleneck, 3 decoding)	90.2 ± 1.3	75.1 ± 2.1	75.3 ± 1.9	83.6 ± 3.5	88.0 ± 0.9	90.9 ± 0.8	87.5 ± 0.3	84.4

Table 1. Ablation on different numbers of interaction blocks on the Physion dataset with the contact prediction accuracy (%)

	Dominoes	Contain	Link	Support	Drop	Collide	Roll	Average
DPI [5]	82.3 ± 1.3	72.3 ± 1.8	63.7 ± 2.2	64.8 ± 2.0	70.7 ± 0.8	84.4 ± 0.7	82.3 ± 0.6	74.4
PointConv U-Net w/o object-centric PointConv	$\mathbf{88.2} \pm 0.3$	71.8 ± 3.5	70.0 ± 4.3	$\mathbf{82.4} \pm 2.5$	82.7 ± 1.5	$\mathbf{90.9} \pm 0.3$	$\mathbf{87.8} \pm 0.3$	**82.0**
PointConv U-Net	$\mathbf{90.2} \pm 1.3$	75.1 ± 2.1	75.3 ± 1.9	$\mathbf{83.6} \pm 3.5$	$\mathbf{88.0} \pm 0.9$	$\mathbf{90.9} \pm 0.8$	$\mathbf{87.5} \pm 0.3$	**84.4**

Table 2. Ablation on the interaction block design that creates the separation between object and relational PointConv

(e.g., more interaction blocks if longer-range force propagation needs to be modeled). The result with 3 blocks corresponds to the scenario where a single downsampling and upsampling step is employed. One can see that results for most scenarios in the Physion dataset do not change significantly with further downsampling.

In Table 2, we compare our approach with an alternative where the entire point cloud is directly processed without separate object and relational PointConv layers. Note that this baseline requires an expensive KNN neighbor search every frame across all PointConv layers, whereas the neighbor search needs to be done only once when the object PointConv layers are used for rigid body objects. Results in Table 2 show that the non-object-centric baseline can still perform well, especially for scenes such as Roll and Collide, where objects are well separated. This result makes sense, as when objects are well separated most of the time, the KNN neighbors generally come from the same object except for a few collision events. Also, note that our non-object-centric baseline generally performs better than other non-object-centric baselines such as DPI [5].

References

[1] Kelsey R Allen, Yulia Rubanova, Tatiana Lopez-Guevara, William F Whitney, Alvaro Sanchez-Gonzalez, Peter Battaglia, and Tobias Pfaff. Learning rigid dynamics with face interaction graph networks. In *The Eleventh International Conference on Learning Representations*, 2023. 1

[2] Daniel Bear, Elias Wang, Damian Mrowca, Felix Binder, Hsiao-Yu Tung, Pramod RT, Cameron Holdaway, Sirui Tao, Kevin Smith, Fan-Yun Sun, Fei-Fei Li, Nancy Kanwisher, Josh Tenenbaum, Dan Yamins, and Judith Fan. Physion: Evaluating physical prediction from vision in humans and machines. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Curran, 2021. 1

[3] Artur Grigorev, Michael J Black, and Otmar Hilliges. Hood: Hierarchical graphs for generalized modelling of clothing dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16965–16974, 2023. 1

[4] Jiaqi Han, Wenbing Huang, Hengbo Ma, Jiachen Li, Josh Tenenbaum, and Chuang Gan. Learning physical dynamics with subequivariant graph neural networks. *Advances in Neural Information Processing Systems*, 35:26256–26268, 2022. 1

[5] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *International Conference on Learning Representations*, 2018. 2